热敏感应温度计的使用

文件编码：**HA0098S**

简介

本套装置是采用 HT49R50 的 PC.0，PC.1，PC.2 引脚判读热敏电阻的变化数，以达到温度感测的功能。本套装置最终为带一位小数的 LCD 显示，允许测量的温度范围为（-40°C~100°C），误差为±1°C。本装置仅为参考范例，在高温和低温时，参考电阻阻值会受温度影响而变化，在使用时，应作相应修正。

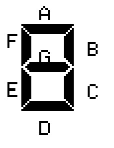
使用说明

1. 新建一个 PROJECT,把 TEMP.ASM 和 CALCULATE.ASM 同时加载到项目下（[Project/Edit]路径）。
2. 温度显示为十进制数 LCD 显示，最终的显示形式如下图一所示。第一个模块 G 段为正负温度符号显示，当所测温度为负温时即点亮，正温时即熄灭；第二个模块为温度十位值显示；第三个模块为温度个位值显示；第四个模块为温度小数字值显示；当温度值超过测量范围（-40°C~100°C）时，第二，三，四个模块分别只会点亮 G 段。SEG 口和 COM 口所对应的连接如图二所示，使用时请特别注意对应连接。
3. 由于所使用的热敏电阻型号和规格不尽相同，请注意更新程序中的 TABLE_TEMPERATURE 表格，由于程序设定，表格应将各温度下的热敏电阻值放大 100 倍后，从大到小键入；主程序 temp.asm 档案中所定义的参数 TEMP_START_ADDR，TEMP_END_ADDR 和 NUM 也要根据所建的²热敏电阻-温度分度表²作相应修改：

```
#DEFINE   TEMP_START_ADDR     0F0AH
#DEFINE   TEMP_END_ADDR       TEMP_START_ADDR+100-(-40)
#DEFINE   NUM                 TEMP_START_ADDR-(-40)
```

注意: TEMP_START_ADDR：TABLE_TEMPERATURE 表格首地址。
TEMP_END_ADDR：TABLE_TEMPERATURE 表格末地址。
NUM：可量测的负温度的范围。



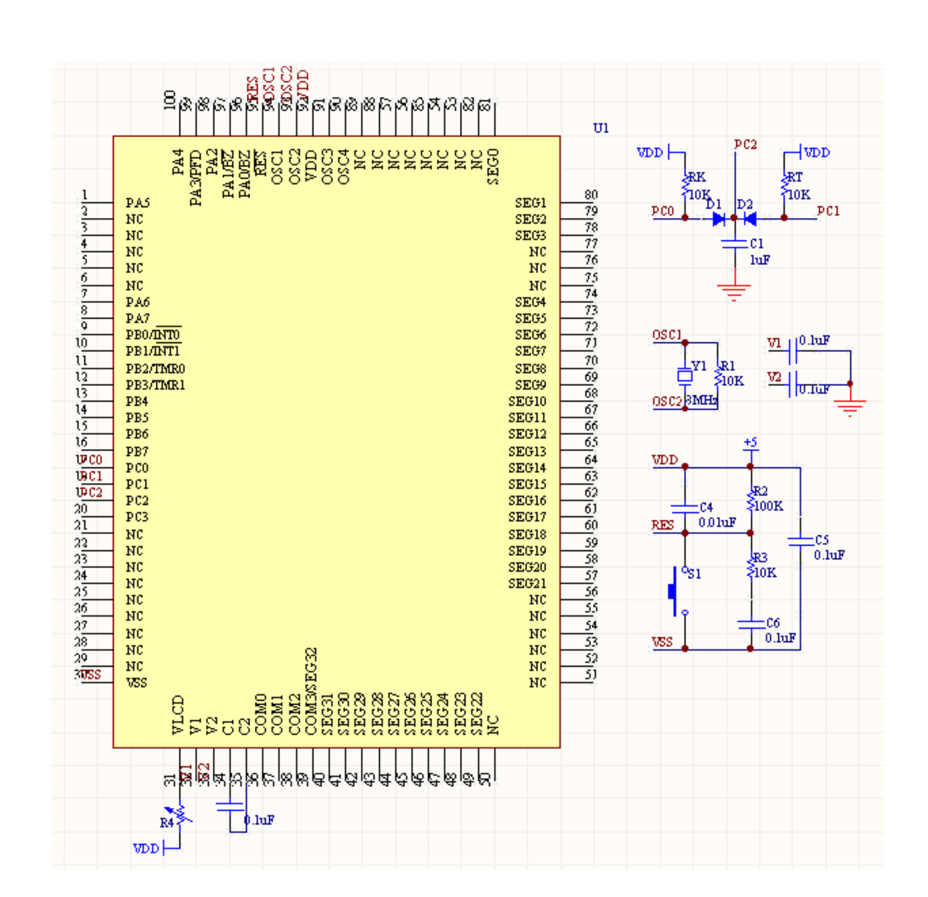


图一

	SEG0	SEG1	SEG2	SEG3	SEG4	SEG5	SEG6	SEG7	- - - -	SEG31
COM0	2A	2E	3A	3E	4A	4E	1G	—	- - - -	—
COM1	2B	2F	3B	3F	4B	4F	DOT	—	- - - -	—
COM2	2C	2G	3C	3G	4C	4G	—	—	- - - -	—
COM3	2D	—	3D	—	4D	—	—	—	- - - -	—

表一

电路图



原理说明

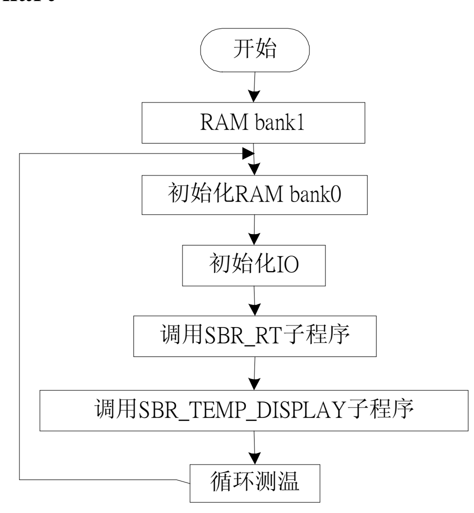
以上原理图中 RK 为 10K 的精密电阻；RT 为 10K 的热敏电阻；C1 为 1mF 的电容。其工作原理为：在 C1 放电完全后，只用参考电阻 RK 给 C1 充电，设充电至 PC.2 检测到高准位时,计数器定时为 T1；再将 C1 放电完全，改用热敏电阻 RT 给 C1 充电，充电至 PC.2 检测到高准位时，计数器定时为 T2。

从电容电压对应公式 VC=V0(1−e−T/RC)，可得：RT=T2×RK/T1。

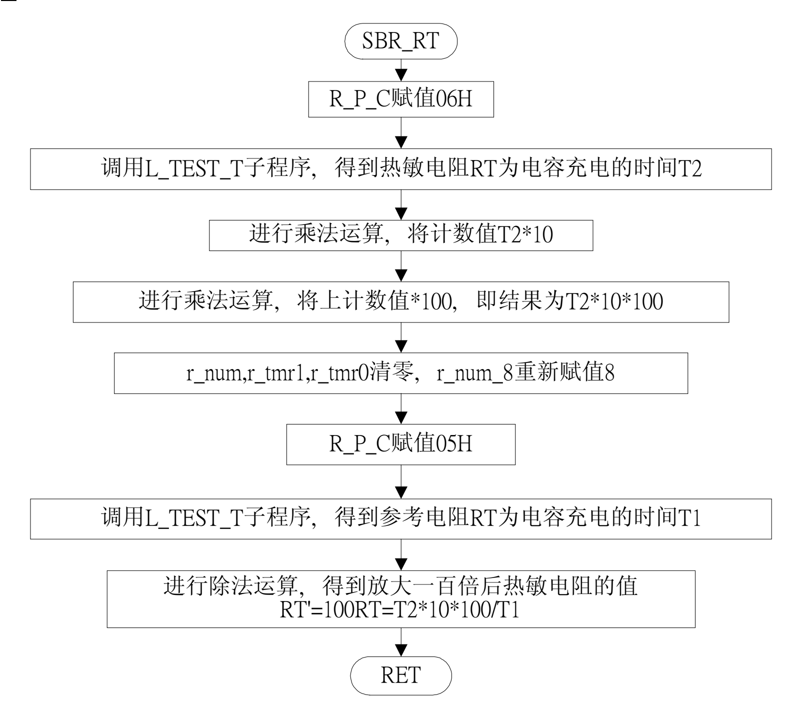
通过程序设置可以得到热敏电阻 RT 的值，并通过查表可以得到温度值。从上述可得，该温测电路的误差主要来自：微控制器的定时器精度、RK 电阻的精度以及热敏电阻 RT 的精度。程序中也设置多次充电，取充电时间平均值以减小误差。

流程图

MAIN Flow Chart



∈SBR_RT Subroutine Flow Chart



SBR_TEMP_DISPLAY Subroutine Flow Chart

SBR_TEMP_DISPLAY

TBLP指向温度表格的首地址

读取温度表格值高低位分别保存于(R_TBLH,R_TABLE_L)

(R_TBLH, R_TABLE_L) = (R_RT_H,R_RT_L)?

Y

N

TBLP->R_TBLP

标志位FLAG_EQU置1, 小数位R_DOT清零

标志 FLAG_EQU 清零

计算(R_TBLH, R_TABLE_L)-(R_RT_H,R_RT_L)的值

C < 0 ?

Y

N

TBLP=TEMP_START_ADDR?

N

Y

TBLP=TEMP_END_ADDR?

Y

N

计算(R_RT_H,R_RT_L)-(R_TBLH, R_TABLE_L)

保存TO0->R_NUM0 TO1->R_NUM1

计算(R_NUM1,R_NUM0)+(R_TO1, R_TO0)=(R_SUM1,R_SUM0)

SET BP.1, 40H-46H清零

41H->MP1, SET R1.2 温度计十位显示?

MP1+2->MP1, SET R1.2 温度计个位显示?

MP1+2->MP1, SET R1.2 温度计小数位显示?

CLR BP

RET

TO0->R_TO0
TO1->R_TO1
TBLP->R_TBLP

TBLP+1

(R_RT_H,R_RT_l)=(ZERO_H,ZERO_L)?

Y

N

不显示负温符号'-'

(R_RT_H,R_RT_l)<(ZERO_H,ZERO_L)?

Y

N

显示负温符号'-'

FLAG_EQU=1?

N

Y

(R_NUM1,R_NUM0)->(data1,data0)调用L_DOT

TBLP->R_TBLP

NUM-R_TBLP->R_TBLP

不显示负温符号'-'

FLAG_EQU=1?

N

Y

(R_TO1,R_TO0)->(data1,data0),调用L_DOT

R_TBLP=TEMP_END_ADDR?

N

Y

R_TBLP-NUM->R_TBLP

JMP L_16_to_10

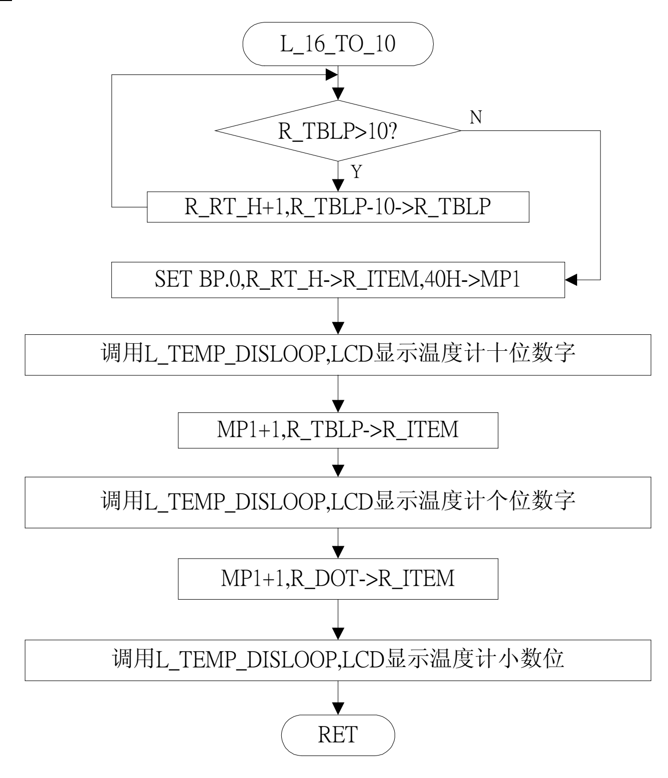
9->R_ITEM, 40H->MP1 调用L_TEMP_DISLOOP, 温度计十位显示9

MP1+1->MP1,L_TEMP_DISLOOP, 温度计个位显示9

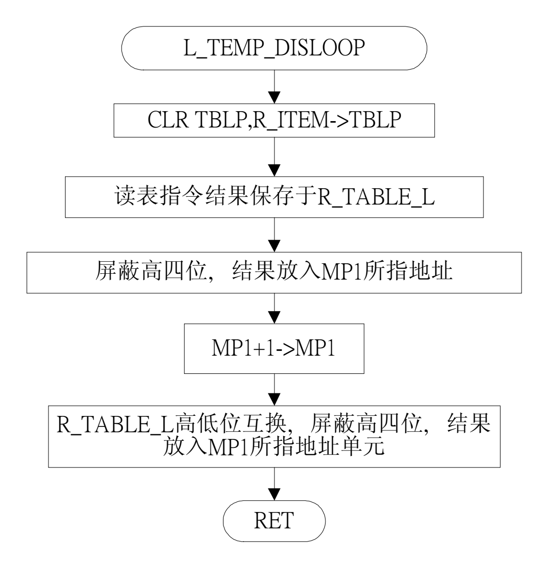
MP1+1->MP1,调用L_TEMP_DISLOOP, 温度计小数位显示9

RET

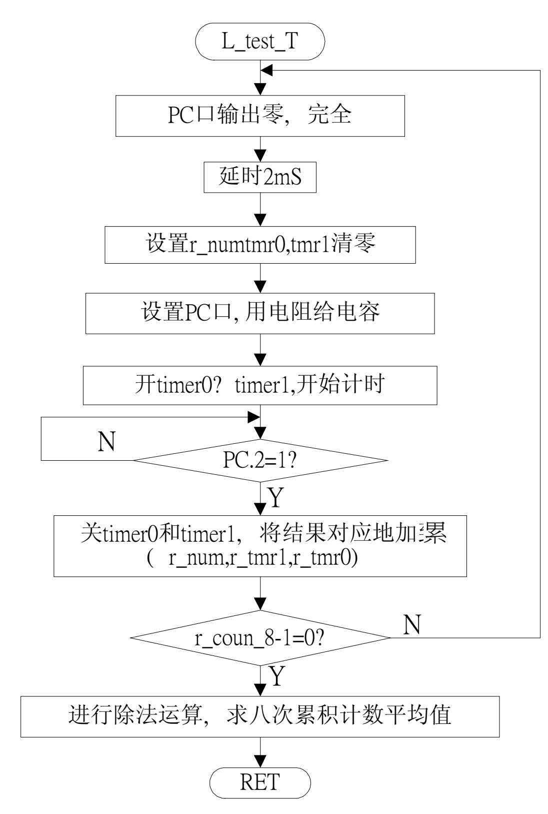
L_16_TO_10 Label Flow Chart



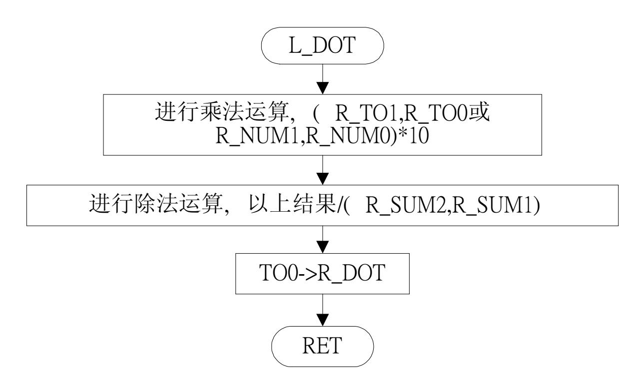
L_TEMP_DISLOOP Subroutine Flow Chart



L_TEST_T Subroutine Flow Chart



L_DOT Subroutine Flow Chart



程序范例

```
******************File Name: Temp.Asm(主程序)******************
;;Client:
;;ID Code:
;;HT_ICE Version:
;;HT_IDE Version: V6.6
;;Program Name: Temp.Prj
;;Program Version:
;;Established Date: 2005/08/11
;;Programmer: Song
;;--------------------------------------------
;;Main Function: Test Temperature
;;MCU Body:Ht49r50
;;VDD: 5V
;;MCU Frequency: 8MHz
;;--------------------------------------------
;;Mask Option:
;Vdd : 5.00V
;OSC: Crystal
;Fsys: 8MHz
;Package: 100 QFP-A
;Wdt: Disable
;Clr Wdt: One Clear Instruction
;Wake_Up PA0-7: Non Wake Up
;Pull-high PA: Nmos&PULL-HIGH
;Pull-high PC: Nmos&NON-PULL-HIGH
;Timer0 clock source: System clock
;Timer1 clock source: Timer0 overflow
;TMR0&TMR1 PFD: Disable
;Clock source: WDT OSC(12K)
;BZ/BZB: Disable
;LCD duty: 1/4 duty
;LVR: Disable
;LVD: Disable
;;=================================================
#INCLUDE        HT49R50A-1.INC
#INCLUDE        MACRO.ASM
#INCLUDE        CALCULATE.INC
;;=================================================
#DEFINE         P_A       PA
#DEFINE         P_B       PB
#DEFINE         P_C       PC
#DEFINE         ZERO_H0BH                 ;;the high byte of RT as T=0 degree
#DEFINE         ZERO_L 3AH                ;;the low byte of RT as T=0 degree
#DEFINE         TEMP_START_ADDR           0F0AH
                                          ;;the start address of temperature
                                          ;;table
```

```
#DEFINE      TEMP_END_ADDR       TEMP_START_ADDR+100-(-40)
                                 ;;the end address of temperature table
                                 ;;100:the highest temperature tested;
#DEFINE      NUM      TEMP_START_ADDR-(-40)
                                 ;;-40:the lowest temperature tested
;;=================================================
DATA.SECTION'DATA'
ACCBAK       DB       ?          ;;save acc registor value
STATUSBAK    DB       ?          ;;save status registor value
FLAG_EQU     DBIT                ;;compare temperature flag
R_P_C        DB       ?          ;;rigistro for pc
BUF1         DB       ?          ;;registor for delay
BUF2         DB       ?          ;;registor for delay
BUF3         DB       ?          ;;registor for delay
R_TBLP       DB       ?          ;;save tblp    value
R_TBLH       DB       ?          ;;save the high byte of table code
R_TABLE_L    DB       ?          ;;save the low byte of rom table code
R_RT_L       DB       ?          ;;the low byte of RT summation & RT
R_RT_H       DB       ?          ;;the high byte of RT summation & RT
                                 ;;& temperature
R_RT         DB       ?          ;;the (16~23)bit of RT summation
R_ITEM       DB       ?          ;;registor for diaplay subroutine
R_TO0        DB       ?          ;;save to0
R_TO1        DB       ?          ;;save to1
R_TO2        DB       ?          ;;save to2
R_TO3        DB       ?          ;;save to3
R_COMPARE    DB       ?          ;;for RT compare
R_NUM                 DB         ? ;;for RT calculate&interrupt time
R_NUM1       DB       ?          ;;for RT calculate
R_NUM2       DB       ?          ;;for RT calculate
R_SUM1       DB       ?          ;;for RT calculate
R_SUM2       DB       ?          ;;for RT calculate
R_DOT                 DB         ? ;;the decimal of temperature
R_COUN_8     DB       ?          ;;for 8 times cycle
R_COUN_4     DB       ?          ;;for 4 times cycle
R_TMR0       DB       ?          ;;the low byte of charging time
                                 ;;summation
R_TMR1       DB       ?          ;;the high byte of charging time
                                 ;;summation
RC_NUM       DB       ?          ;;the (16~23)bite of charging time
                                 ;;summation
```

```
MAIN.SECTION    AT    0    'CODE'
        JMP                 MAIN

ORG             04H                         ;;external int0 interrupt vector
        JMP                 ISR_EXTINT0

ORG             08H                         ;;external int1 interrupt vector
        JMP                 ISR_EXTINT1

ORG             0CH                         ;;timer0 interrupt vector
        JMP                 ISR_TMR0

ORG             10H                         ;;timer1 interrupt vector
        JMP                 ISR_TMR1

ORG             14H                         ;;timebase interrupt vector
        JMP                 ISR_TIMEBASE

ORG             18H                         ;;rtc interrupt vector
        JMP                 ISR_RTCC
;;==========================================================
;;==========================================================
MAIN:
        CALL    INI_LCD_RAM                 ;;initial bank1 of RAM
        CALL    INI_RAM                     ;;initial band0 of RAM
        CALL    INI_IO                      ;;initial I/O
        CALL    SBR_RT                      ;;get the value of RT
        CALL    SBR_TEMP_DISPLAY            ;;display the temperature with LCD
        JMP             $-4                 ;;run again
;;=====================================================
ISR_EXTINT0:            ;;(no used)04 interrupt
        PUSH
        POP
        RETI
;;=====================================================
ISR_EXTINT1:            ;;(no used)08 interrupt
        PUSH
        POP
        RETI
;;===================================================
ISR_TMR0:                                   ;;(no used)0c interrupt
        PUSH
        POP
        RETI
```

```
;;=====================================================
ISR_TMR1:                               ;;10 interrupt
        PUSH
        INC     R_NUM
        POP
        RETI
;;=====================================================
ISR_TIMEBASE:                           ;;(no used)14 interrupt
        PUSH
        POP
        RETI
;;=====================================================
ISR_RTCC:                               ;;(no used)18 interrupt
        PUSH
        POP
        RETI
;;=====================================================
#INCLUDE        SUBROUTINE.ASM

***********File Name: Subroutine .Asm（子程序）***************
;;----------INI_RAM: initial bank 0 of ram------------------
INI_RAM:                                ;;initial bank 0 of RAM
        CLR     BP.0                    ;;bp point to bank 0 of RAM
        XMOV    MP0,    60H             ;;ram start address 60H
        CLR     R0
        INC     MP0
        SZ      MP0                     ;;ram end address FFH
        JMP     $-3

        XMOV    INTC0,  01H             ;;enable gloable interrupt
        XMOV    INTC1,  01H             ;;enable timer1 interrupt
        XMOV    TMR0C, 0A0H             ;;timer0 mode:timer mode and select
                                        ;;system clock source
        XMOV    TMR1C, 80H              ;;timer1 mode:timer mode&mask option
                                        ;;clock source
        XMOV    R_COUN_8,8
        XMOV    R_COUN_4,4
        RET
```

```
;------------INI_LCD_RAM:initial bank 1 of RAM--------------
INI_LCD_RAM:            ;;initial bank 1 of RAM
        SET     BP.0                    ;;bp point to bank 1 of RAM
        XMOV    MP1,    40H             ;;ram start address 40H
        CLR     R1
        INC     MP1
        EJMP    MP1,    60H             ;;ram end address 60H
        JMP     $-5
        XMOV    MP1,    46H             ;;lighten radix point[46H].1
                                        ;;(seg6&com1)
        SET     R1.1
        CLR     BP
        RET
;;-------------INI_LCD_RAM:initial bank 1 of RAM---------------
INI_IO:                                 ;;initial io
        SET     P_A                     ;;initial i/o input port
        SET     P_B
        SET     P_C
        RET
;;--------------SBR_RT:get the value of RT---------------------
SBR_RT:                                 ;;get the value of RT
        XMOV    R_P_C,  06H             ;;charge with pc.1
        CALL    L_TEST_T                ;;test charge time(T2)
        XMOV    DATA0,  TO0             ;;the average charge time(T2)
                                        ;;(to2 to1 to0)
        XMOV    DATA1,  TO1
        XMOV    DATA2,  TO2             ;;multiplicand(data2 data1 data0)
        XMOV    DATA4,  0AH             ;;multiplier(data6 data5 data4)
        CLR     DATA5
        CLR     DATA6
        CALL    UNBIN_MUL_24            ;;T2×RK(10K)
        XMOV    DATA0,  TO0             ;;the result of multiplication
                                        ;;(to3 to2 to1 to0)
        XMOV    DATA1,  TO1
        XMOV    DATA2,  TO2
        XMOV    DATA3,  TO3             ;;multiplicand
                                        ;;(data3 data2 data1 data0)

        XMOV    DATA4,  64H             ;;multiplier(data7 data6 data5 data4)
        CLR     DATA5
        CLR     DATA6
        CLR     DATA7
        CALL    UNBIN_MUL_32            ;;T2*10*100
        XMOV    R_TO0,  TO0             ;;the result of multiplication
                                        ;;(to3 to2 to1 to0)
        XMOV    R_TO1,  TO1             ;;save
        XMOV    R_TO2,  TO2
        XMOV    R_TO3,  TO3
```

```
;;=================================================
        CLR     R_TMR0
        CLR     R_TMR1
        CLR     RC_NUM                  ;;clear for next use
        XMOV    R_COUN_8,8
L_AVER4:
        XMOV    R_P_C,  05H             ;;charge with pc.0
        CALL    L_TEST_T                ;test charge time(T1)

        XMOV    DATA4,  TO0             ;;the average charge time(T1)
                                        ;;(to2 to1 to0)
        XMOV    DATA5,  TO1             ;;divisor(data4 data5 data6 data7)
        XMOV    DATA6,  TO2
        CLR     DATA7
        XMOV    DATA0,  R_TO0           ;;dividend(data3 data2 data1 data0)
        XMOV    DATA1,  R_TO1
        XMOV    DATA2,  R_TO2
        XMOV    DATA3,  R_TO3

        CALL    UNBIN_DIV_32            ;;devision T2×10×100/T1=RT×100
        XADDM   R_RT_L,TO0              ;;summation of RT×100
        XADCM   R_RT_H,TO1
        XADCM   R_RT,   TO3
        XMOV    R_COUN_8,8              ;;for next use
        CLR     R_TMR0
        CLR     R_TMR1
        CLR     RC_NUM                  ;;clear for next use
        SDZ     R_COUN_4                ;;for 4 times
        JMP     SBR_RT
        RRC     R_RT                    ;;/2
        RRC     R_RT_H                  ;;/2
        RRC     R_RT_L                  ;;/2
        CLR     C
        RRC     R_RT                    ;;/2
        RRC     R_RT_H                  ;;/2
        RRC     R_RT_L                  ;;/2 (r_rt_h,r_rt_l)=RT×100=RT'
        RET
;;=================================================
L_TEST_T:                               ;;test charge time
        XMOV    P_C,    00H             ;;discharge completely
        CALL    DELAY_2mS               ;;delay 2mS

        CLR     R_NUM
        SET     TMR1C.4
        CLR     TMR1C.4
        CLR     TMR0                    ;;clear timer0&timer1 preload
                                        ;;registor
        CLR     TMR1
        XMOV    P_C,    R_P_C           ;;charge with PC.x port
```

```
        SET     TMR0C.4                 ;;timer0 on
        SET     TMR1C.4                 ;;timer1 on
        MOV     P_C,    A
        SNZ     P_C.2                   ;;wait for charge finished
        JMP     $-2

        CLR     TMR0C.4                 ;;timer0 off
        CLR     TMR1C.4                 ;;timer1 off
        CLR     C
        XADDM R_TMR0,TMR0               ;;the summation of time
        XADCM R_TMR1,TMR1
        XADCM RC_NUM,R_NUM
        SDZ     R_COUN_8                ;;for 8 times
        JMP     L_TEST_T
        XMOV    DATA0,  R_TMR0          ;;dividend(data2 data1 data0)
        XMOV    DATA1,  R_TMR1
        XMOV    DATA2,  RC_NUM
        XMOV    DATA4,  8               ;;divisor(data4 data5 data6)
        CLR     DATA5
        CLR     DATA6
        CALL    UNBIN_DIV_24            ;;division
        RET
;;--SBR_TEMP_DISPLAY:chart table to get and display the temperature--
SBR_TEMP_DISPLAY:                       ;;chart table to get and display the
                                        ;;temperature
        CLR     BP
        XMOV    TBLP,   TEMP_START_ADDR
                                        ;;tblp point to the first address of
                                        ;;temperature table
L_TABLE_COMPARE:
        TABRDLR_TABLE_L                 ;;low byte of table code→r_table_l
        XMOV    R_TBLH,TBLH             ;;high byte of table code→r_tblh
        EJMP    TBLH,   R_RT_H          ;;if RT′ equals to the value of table
        JMP     L_NEQU
        EJMP    R_TABLE_L,R_RT_L
        JMP     L_NEQU
        XMOV    R_TBLP,TBLP
        JMP     L_EQU
L_EQU:                                  ;;when RT equals to the value of table
        SET     FLAG_EQU                ;;set equal flag
        CLR     R_DOT
        JMP     L_ZERO_COMP
L_NEQU:                                 ;;if unequal
        CLR     FLAG_EQU                ;;clr equal flag
        XMOV    DATA0,  R_TABLE_L       ;;minuend
        XMOV    DATA1,  R_TBLH
        XMOV    DATA4,  R_RT_L          ;;subtrahend
        XMOV    DATA5,  R_RT_H
        CALL    UNBIN_SUB_16            ;;compare table value with RT
```

```
        SZ      C
        JMP     L_NEXT                  ;;if bigger than RT
        NEJMP   TBLP,    TEMP_START_ADDR
                                        ;;if less than RT&if tblp point to
                                        ;;the biggest RT
        JMP     L_TEMP_OVER             ;;if over the range of table
        JMP     L_GET_TEMP              ;;if not

L_NEXT:                                 ;;tblp point to the next address
        NEJMP   TBLP,    TEMP_END_ADDR
                                        ;;if tblp point to the smallest RT
        JMP     L_TEMP_OVER             ;;if excess the range of table
        XMOV    R_TO0,   TO0            ;;if not and save as r_to0
        XMOV    R_TO1,   TO1            ;;save as r_to1
        XMOV    R_TBLP,TBLP             ;;save tblp
        INC     TBLP                    ;;point to the next address
        JMP     L_TABLE_COMPARE         ;;compare again

L_GET_TEMP:                             ;;figure out the temperature
        XMOV    DATA0,  R_RT_L          ;;minuend(data1 data0)
        XMOV    DATA1,  R_RT_H
        XMOV    DATA4,  R_TABLE_L       ;;subtrahend(data5 data4)
        XMOV    DATA5,  R_TBLH
        CALL    UNBIN_SUB_16            ;;subtration
        XMOV    R_NUM1, TO0             ;;save as r_num1
        XMOV    R_NUM2, TO1             ;;save as r_num2

        XMOV    DATA0,  R_NUM1          ;;augend(data1 data0)
        XMOV    DATA1,  R_NUM2
        XMOV    DATA4,  R_TO0           ;;addend(data5 data4)
        XMOV    DATA5,  R_TO1
        CALL    UNBIN_ADD_16            ;;addition(r_num2,r_num1)
                                        ;;+(r_to1,r_to0)
        XMOV    R_SUM1,TO0
        XMOV    R_SUM2,TO1              ;;(r_num2,r_num1)+(r_to1,r_to0)
                                        ;;=(r_sum2,r_sum1)
L_ZERO_COMP:                            ;;compare with zero
        EJMP    R_RT_H,ZERO_H
        JMP     L_UNEQU_ZERO
        EJMP    R_RT_L,ZERO_L
        JMP     L_UNEQU_ZERO
L_EQU_ZERO:                             ;;equal to zero
        SET     BP.0
        XMOV    MP1,     46H
        CLR     R1.0                    ;;unlighen the sign '-' ([46H].0)
        JMP     L_OVER_LOOP
```

```
L_UNEQU_ZERO:                         ;;uneaual to zero
        XMOV    DATA0,  R_RT_L
        XMOV    DATA1,  R_RT_H
        XMOV    DATA4,  ZERO_L
        XMOV    DATA5,  ZERO_H
        CALL    UNBIN_SUB_16          ;;RT compares with 0b3a
                                      ;;(T vs. 0 degree)
        SZ      C
        JMP     L_LOW_ZERO            ;;if lower than zero
        JMP     L_OVER_ZERO           ;;if higher than zero

L_LOW_ZERO:                           ;;if lower than zero
        SET     BP.0
        XMOV    MP1,    46H
        SET     R1.0                  ;;lighen the sign '-' ([46h].0)
        CLR     BP
        SZ      FLAG_EQU              ;;judge if or not integer
        JMP     L_LOW_lOOP            ;;if integer
        XMOV    DATA0,  R_NUM1
        XMOV    DATA1,  R_NUM2
        CALL    L_DOT                 ;;if not ,then deal with
                                      ;;the decimal(T<0)
        XMOV    R_TBLP,TBLP           ;;if T<0 degree,then the address
                                      ;;is the smaller one
L_LOW_lOOP:                           ;;T<0 degree,then [T]=NUM-r_tblp
        MOV     A,      NUM
        SUB     A,      R_TBLP
        MOV     R_TBLP,A
        CLR     R_RT_H                ;;for next user
        JMP     L_16_to_10

L_OVER_ZERO:                          ;;if higer than zero
        SET     BP.0
        XMOV    MP1,    46H
        CLR     R1.0                  ;;unlighen the sign '-' ([46h].0)
        CLR     BP
        SZ      FLAG_EQU              ;;judge if or not integer
        JMP     L_OVER_LOOP           ;;if integer
        XMOV    DATA0,  R_TO0
        XMOV    DATA1,  R_TO1
        CALL    L_DOT                 ;;if not,then deal with
                                      ;;the decimal(T>0)
L_OVER_LOOP:                          ;;T>0 degree,then [T]=r_tblp-NUM
        NEJMP   R_TBLP,TEMP_END_ADDR
                                      ;;check if the temperature
                                      ;;is 100 degree
        JMP     L_TEMP_100            ;;if 100,then jump
        MOV     A,      R_TBLP        ;;if not ,[T]=r_tblp-NUM
        SUB     A,      NUM
```

```
        MOV     R_TBLP,A
        CLR     R_RT_H
L_16_TO_10:                             ;;convert T from hex to decimal
        LBERJ   R_TBLP,0AH              ;;r_tblp-0ah
        JMP     $+4                     ;;if r_tblp-0ah<0
        INC     R_RT_H                  ;;if r_tblp-0ah>0
        MOV     R_TBLP,A
        JMP     L_16_to_10              ;;(T)H={r_rt_h,r_tblp}D
        SET     BP.0
        XMOV    R_ITEM,R_RT_H           ;;the number displayed->r_item
        XMOV    MP1,    40H             ;;the adress displayed
        CALL    SBR_TEMP_DISLOOP        ;;display
        INC     MP1                     ;;the adress displayed
        XMOV    R_ITEM,R_TBLP           ;;the number displayed->r_item
        CALL    SBR_TEMP_DISLOOP        ;;display
        INC     MP1                     ;;the adress displayed
        XMOV    R_ITEM,R_DOT            ;;the number displayed->r_item
        CALL    SBR_TEMP_DISLOOP        ;;display
        CLR     BP
        RET                             ;;return

L_TEMP_100:                             ;;if T=100,then display 99.9
        SET     BP.0
        XMOV    R_ITEM,9
        XMOV    MP1,    40H
        CALL    SBR_TEMP_DISLOOP        ;;display 9
        INC     MP1
        CALL    SBR_TEMP_DISLOOP        ;;display 9
        INC     MP1
        CALL    SBR_TEMP_DISLOOP        ;;display 9
        CLR     BP
        RET
;;================================================
L_TEMP_OVER:                            ;;if over the testing range,
                                        ;;then display '---'
        SET     BP.0
        XMOV    MP1,    40H
L4:     CLR     R1                      ;;clear [40h]~[46h]
        INC     MP1
        EJMP    MP1,    46H
        JMP     L4
        XMOV    MP1,    41H             ;;display '---'
        SET     R1.2
        INC     MP1
        INC     MP1
        SET     R1.2
        INC     MP1
        INC     MP1
        SET     R1.2
```

```
        CLR     BP
        RET
;;====================================================
L_DOT:                                  ;;deal with the decimal(num*10/sum)
        XMOV    DATA4,  0AH             ;;*10
        CLR     DATA5
        CALL    UNBIN_MUL_16
        XMOV    DATA0,  TO0
        XMOV    DATA1,  TO1
        XMOV    DATA4,  R_SUM1
        XMOV    DATA5,  R_SUM2
        CALL    UNBIN_DIV_16            ;;/(sum2,sum1)
        XMOV    R_DOT,  TO0             ;;save as r_dot
        RET
;;=================================================
SBR_TEMP_DISLOOP:                       ;;display a pointed number at
                                        ;;a pointed address
        CLR             TBLP            ;;mp1->the pointed address
        XADDM TBLP,     R_ITEM          ;;number displayed->r_item
        TABRDLR_TABLE_L
        XAND    R_TABLE_L,0FH
        MOV     R1,     A               ;;display the number r_item saved
        INC     MP1
        SWAP    R_TABLE_L
        XAND    R_TABLE_L,0FH
        MOV     R1,     A
        RET
;;====================================================
DELAY_2mS:                              ;;delay about 2mS
        XMOV    BUF2,   64              ;;recycle for 64 times
        SET     BUF1                    ;;32uSfor one time
        SDZ     BUF1
        JMP     $-1
        SDZ     BUF2
        JMP     $-4
        RET
;;------------TABLE OF NUMBER AND TEMPERATURE----------------
TABLE0 .SECTION AT 0F00H 'CODE'         ;;the displayed code of number0~9
DC      03FH, 006H,05BH,04FH            ;;0,1,2,3
DC      066H, 06DH,07DH,007H            ;;4,5,6,7
DC      07FH, 06FH                      ;;8,9
TABLE_TEMPERATURE.SECTION AT 0F0AH 'CODE'
DC      22668,21384,20183,19058,18005,17018,16093,15225,14411,13646
;;-40~-31
DC      12928,12232,11578,10964,10387,09845,09335,08855,08404,07979
;;-30~-21
DC      07578,07196,06835,06496,06175,05873,05588,05318,05064,04823
;;-20~-11
```

```
DC      04596,04378,04172,03977,03793,03619,03454,03297,03148,03008
;;-10~-9
DC      02874,02746,02624,02509,02399,02295,02196,02102,02013,01928
;;0~9
DC      01848,01770,01696,01626,01559,01495,01435,01377,01322,01269
;;10~19
DC      01219,01171,01125,01082,01040,01000,00962,00925,00890,00856
;;20~29
DC      00824,00793,00764,00735,00708,00683,00658,00634,00612,00590
;;30~39
DC      00569,00549,00530,00511,00494,00477,00460,00447,00430,00415
;;40~49
DC      00402,00388,00375,00362,00350,00339,00328,00317,00307,00297
;;50~59
DC      00288,00278,00270,00261,00253,00245,00238,00230,00223,00217
;;60~69
DC      00210,00204,00198,00192,00186,00181,00175,00170,00165,00161
70~79
DC      00156,00152,00147,00143,00139,00135,00131,00128,00124,00121
;;80~89
DC      00118,00113,00111,00108,00105,00103,00100,00097,00095,00092
;;90~99
DC      00090
;;100
```